#### 导航

博客园

首 页

新随笔

联系

订阅 🔤

管 理

# Section 1 Company of the company of

29 30 31 1 2 3 4

#### 公告

#### 不管做什么,只要坚持下去就 会看到不一样!在路上,不卑 不亢!

我的CSDN博客 我的博客

昵称: 阿飞(dufyun) 园龄: 2年1个月 粉丝: 29 关注: 3 +加关注

找找看

#### 搜索

# 常用链接

我的随笔

我的评论

我的参与

最新评论 我的标签

#### 最新随笔

- 1. Mysql索引整理总结
- 2. jstat命令总结
- 3. Java死锁排查和Java CP
- U 100% 排查的步骤整理
- 4. Spring-Session实现Ses sion共享实现原理以及源码解析
- 5. Spring-Session实现Ses sion共享Redis集群方式配置 教程
- 6. Spring-Session实现Ses sion共享入门教程
- 7. Redis高可用集群-哨兵模式 (Redis-Sentinel) 搭建
- 配置教程【Windows环境】 8. Redis创建高可用集群教
- 程【Windows环境】
- 9. Redis集群主从复制(一 主两从)搭建配置教程【Win dows环境】
- 10. 【教程】Tomcat 的cat alina.out 日志按照自定义日期格式进行切割

### 我的标签

java集合系列(10) Quartz(5) Redis(5) redis安装(5)

# SSM框架——Spring+SpringMVC+Mybatis的搭建教程

#### 一: 概述

SSM框架在项目开发中经常使用到,相比于SSH框架,它在仅几年的开发中运用的更加广泛。

- Spring作为一个轻量级的框架,有很多的拓展功能,最主要的我们一般项目使用的就是IOC和AOP。
- SpringMVC是Spring实现的一个Web层,相当于Struts的框架,但是比Struts更加灵活和强大!
- Mybatis是 一个持久层的框架,在使用上相比Hibernate更加灵活,可以控制sql的编写,使用 XML或注解进行相关的配置!

根据上面的描述,学习SSM框架就非常的重要了!

#### 二: 搭建一个SSM的过程

1. 使用Maven管理项目

使用Maven在Eclipse中创建一个webapp的项目,具体的创建过程不做演示,如有不会创建的[创建项目] 也可以使用Maven命令进行创建,在Dos窗口进入指定的目录,执行下面命令:

 $\label{local_model} \verb|mvn| | archetype: create -DgroupId= org.ssm.dufy -DartifactId= ssm-demo -Darchetype ArtifactId= maven-archetype-webapp -Dinteractive Mode= false | fal$ 

使用命令要注意,系统安装了Maven,并配置好了环境变量![Maven的安装和环境变量配置]

1. 导入项目(命名创建),添加依赖

导入项目是IDE中,或者直接在IDE创建,一般默认有【src/main/java】,手动创建【src/test/resources】、 【src/test/java】文件夹。

#### 如下项目结构:

- v 📂 ssm
  - > 🎏 src/main/java
  - > @ src/main/resources
  - > 🎏 src/test/java
  - src/test/resources
  - > Maven Dependencies
  - ⇒ Maria > ⇒ JRE System Library [JavaSE-1.7]
  - > 🚵 JUnit 4
  - > 🧁 logs
  - > 🐎 src
  - > 🗁 target

# 然后直接配置 pom.xml文件中的包依赖!

```
instance"
 xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4 0 0.xsd">
 <modelVersion>4.0.0</modelVersion>
 <groupId>org.dufy
 <artifactId>ssm</artifactId>
 <packaging>war</packaging>
 <version>0.0.1-SNAPSHOT
 <name>ssmDemo</name>
 <url>http://maven.apache.org</url>
   <spring.version>4.0.5.RELEASE</spring.version>
   <mybatis.version>3.2.1</mybatis.version>
   <slf4j.version>1.6.6</slf4j.version>
   <log4j.version>1.2.12</log4j.version>
   <mysql.version>5.1.35</mysql.version>
 </properties>
 <dependencies>
 <!-- 添加Spring依赖 -->
   <dependency>
      <groupId>org.springframework
       <artifactId>spring-core</artifactId>
      <version>${spring.version}</version>
   </dependency>
   <dependency>
      <groupId>org.springframework</groupId>
       <artifactId>spring-context</artifactId>
```

shell编程(5) <version>\${spring.version}</version> </dependency> Quartz系列(4) <dependency> Map(4) <groupId>org.springframework</groupId> Vector(4) 集合(4) <version>\${spring.version}</version> TreeMap(3) </dependency> 更多 <dependency> 随笔分类 <groupId>org.springframework</groupId> <artifactId>spring-aop</artifactId> Java(20) <version>\${spring.version}</version> Java Web(5) </dependency> Linux学习(6) <dependency> Mybatis(1) <groupId>org.springframework</groupId> Mysal(3) <artifactId>spring-aspects</artifactId> Nginx(1) <version>\${spring.version}</version> Quartz(8) </dependency> Redis(3) <dependency> Spring&SpringMVC(10) <groupId>org.springframework</groupId> 阿飞的世界(2) <artifactId>spring-tx</artifactId> <version>\${spring.version}</version> 分布式相关(11) 开发工具(3) </dependency> <dependency> 算法 <groupId>org.springframework</groupId> 随笔档案 <artifactId>spring-jdbc</artifactId> <version>\${spring.version}</version> 2018年7月 (2) </dependency> 2018年6月 (1) <dependency> 2018年3月 (8) <groupId>org.springframework</groupId> 2018年1月 (2) <artifactId>spring-web</artifactId> <version>\${spring.version}</version> 2017年10月 (3) </dependency 2017年9月 (1) <!--spring**单元测试依赖 --**> 2017年8月 (1) <dependency> 2017年7月 (10) <groupId>org.springframework</groupId> 2017年6月 (9) <artifactId>spring-test</artifactId> 2017年4月 (7) <version>\${spring.version} <scope>test</scope> 2017年3月 (16) </dependency> 2017年2月 (12) 积分与排名 <!-- spring webmvc相关jar --> <dependency> 积分 - 27261 <groupId>org.springframework</groupId> 排名 - 16137 <artifactId>spring-webmvc</artifactId> <version>\${spring.version} 最新评论 </dependency> 1. Re:Spring-Session实现 <dependency> Session共享实现原理以及源 <groupId>org.springframework</groupId> 码解析 <artifactId>spring-web</artifactId> <version>\${spring.version}</version> @CodeLife-Jerry很高兴帮 </dependency> 助到你!... --阿飞(dufyun) <!-- mysql驱动包 --> 2. Re:Spring-Session实现 <dependency> Session共享实现原理以及源 <groupId>mysql</groupId> 码解析 握草,作为新手的我查了这 <version>\${mysql.version} 么久, 你这一篇完全解决了 </dependency> 我的疑问! 赞! --CodeLife-Jerry <!-- alibaba data source 相关jar包--> 3. Re:Java死锁排查和Java <dependency> <groupId>com.alibaba/groupId> CPU 100% 排查的步骤整理 <artifactId>druid</artifactId> @博客园团队谢谢!... <version>0.2.23 --阿飞-dufyun </dependency> 4. Re:Java死锁排查和Java CPU 100% 排查的步骤整理 <!-- alibaba fastjson 格式化对 --> 您好, 文中的代码块不能高 <dependency> 亮显示,在代码块的```之外 <groupId>com.alibaba 添加空行可解决 <artifactId>fastjson</artifactId> <version>1.1.41 --博客园团队 </dependency> 5. Re:SSM框架一 Spring+SpringMVC+Mybati <!-- logback start --> 的搭建教程 <dependency @阿飞-<groupId>log4j dufyunentity, controller, s <artifactId>log4j</artifactId> ervice这几个包的扫描都要 <version>\${log4j.version}</version> 在applicationContext.xml </dependency> 里配置吗... <dependency> --Youyou\_2608 <groupId>org.slf4j</groupId> 阅读排行榜 <artifactId>slf4j-api</artifactId> <version>\${slf4j.version} 1. SSM框架---

Spring+SpringMVC+Mybati 的搭建教程(77613)

<artifactId>spring-context-support</artifactId> <artifactId>mysql-connector-java</artifactId> </dependency> <dependency> <groupId>ch.qos.logback <artifactId>logback-classic</artifactId>

- 2. 《阿里巴巴Java开发手册》扫描插件正式发布--插件安装和使用分析(4591)
- 3. Quartz学习——Spring 和Quartz集成详解(三) (2887)
- 4. Quartz学习—— SSMM(Spring+SpringMVC+ 和Quartz集成详解(四)(23)
- 5. Quartz学习——Quartz 大致介绍(一)(1991)

#### 评论排行榜

- 1. SSM框架—— Spring+SpringMVC+Mybati 的搭建教程(14)
- 2. 《阿里巴巴Java开发手册》扫描插件正式发布--插件安装和使用分析(8)
- 3. 一张图讲解对象锁和关键 字synchronized修饰方法 (3)
- 4. Java死锁排查和Java CPU 100% 排查的步骤整理 (2)
- 5. Spring-Session实现 Session共享实现原理以及源 码解析(2)

### 推荐排行榜

- 1. SSM框架—— Spring+SpringMVC+Mybati 的搭建教程(2)
- 2. Quartz学习—— SSMM(Spring+SpringMVC+ 和Quartz集成详解(四)(2)
- 3. Quartz学习——Quartz 大致介绍(一)(2)
- 4. 《阿里巴巴Java开发手册》扫描插件正式发布--插件安装和使用分析(1)
- 5. Java死锁排查和Java CPU 100% 排查的步骤整理 (1)

```
<version>1.1.2
  </dependency>
  <dependency>
     <groupId>ch.qos.logback
     <artifactId>logback-core</artifactId>
     <version>1.1.2
  </dependency>
 <dependency>
     <groupId>org.logback-extensions
     <artifactId>logback-ext-spring</artifactId>
     <version>0.1.1
  </dependency>
  <!--mybatis依赖 -->
  <dependency2
     <groupId>org.mybatis
     <artifactId>mvbatis</artifactId>
     <version>${mybatis.version}
  </dependency>
  <!-- mybatis/spring包 -->
  <dependency>
     <groupId>org.mybatis
     <artifactId>mybatis-spring</artifactId>
     <version>1.2.0
 </dependency>
<!-- 添加servlet3.0核心包 -->
       <dependency>
           <groupId>javax.servlet</groupId>
           <artifactId>javax.servlet-api</artifactId>
           <version>3.0.1
       </dependency>
       <dependency>
           <groupId>javax.servlet.jsp</groupId>
           <artifactId>javax.servlet.jsp-api</artifactId>
           <version>2.3.2-b01</version>
       </dependency>
       <!-- jstl -->
       <dependency>
          <groupId>javax.servlet</groupId>
           <artifactId>jstl</artifactId>
          <version>1.2</version>
       </dependency>
  <!--单元测试依赖 -->
 <dependency>
   <groupId>junit</groupId>
   <artifactId>junit</artifactId>
   <version>3.8.1
   <scope>test</scope>
 </dependency>
</dependencies>
<build>
 <finalName>ssmDemo</finalName>
</build>
```

# 1. 创建数据库和表, 生成代码

</project>

创建数据库我参考别人的博客数据库设计,这块没有自己去书写,直接添上代码

### 生成代码请查看:

[Mybatis自动生成代码]

生成的代码导入图片解释:

```
12 User selectByPrimaryKey(Integer id);
记得要改UserMapper.xml 文件
src/main/java
             毋 org.ssm.dufy 本来生成的UserMapper,我改了名字,
                                                                                                                                                                                                                                                                                                                                              int updateByPrimaryKeySelective(User record);
                                                                                                                                                                                                                                                                                                                                              int updateByPrimaryKey(User record);
                           # entity
> ② User.java
                              mapper
                                                                                                                                                                                                                                                                                                                   记得改配置内容中的类全路径! UserMapper 为IuserDao

☑ UserMapper.xml

                                          IUserServiceImpl.java

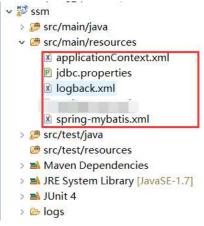
    IUserService.java

                        # web
                                                                                                                                                                                                                                                                                                                                                自己实现的,只写了一个方法做测试,根据id查询user对象
  src/main/resc
src/test/java
    src/test/resources

    Maven Dependencies
    Maven Dipendencies
    Maven Dipende
    ■ JUnit 4
  logs
```

1. Spring 和 mybatis整合,连接数据库,进行Junit测试

将生成的代码拷贝到项目中,然后进行Spring和Mybatis的整合,添加配置文件!



### 主要有

applicationContent.xml: Spring的相关配置! Spring-mhbatis.xml: Spring和Mybatis集成配置!

jdbc.properties:数据库信息配置!

logback.xml:日志輸出信息配置! (不做介绍,详细信息查看源码)

主要介绍applicationContext.xml、Spring-mhbatis.xml、jdbc.properties。主要内容如下:

### jdbc.properties

```
jdbc_driverClassName =com.mysq1.jdbc.Driver
jdbc_url=jdbc:mysq1://localhost:3306/ssm?useUnicode=true&characterEncoding=utf8
jdbc_username=root
jdbc_password=root
```

#### applicationContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</pre>
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
xmlns:context="http://www.springframework.org/schema/context"
xmlns:aop="http://www.springframework.org/schema/aop"
xmlns:tx="http://www.springframework.org/schema/tx"
xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-
context-3.0.xsd
http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-
3.0.xsd
http://www.springframework.org/schema/tx/http://www.springframework.org/schema/tx/spring-tx-3.0.xsd
    <!-- 1.配置jdbc文件 -->
    <bean id="propertyConfigurer"</pre>
class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
           <!-- 2.扫描的包路径, 这里不扫描被@Controller注解的类 --><!--使用<context:component-scan/> 可以不在配置
<context:annotation-config/> -->
    <context:component-scan base-package="org.ssm.dufy">
        <context:exclude-filter type="annotation"</pre>
expression="org.springframework.stereotype.Controller"/>
    </context:component-scan>
    <import resource="classpath:spring-mybatis.xml" />
```

#### spring-mybatis.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</pre>
     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:aop="http://www.springframework.org/schema/aop"
     xmlns:tx="http://www.springframework.org/schema/tx"
xmlns:context="http://www.springframework.org/schema/context"
      xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
    http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop-
3.2.xsd
    \verb|http://www.springframework.org/schema/tx | \verb|http://www.springframework.org/schema/tx/spring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pring-tx-pr
  http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.2.xsd">
<!-- 3.配置数据源 , 使用的alibba的数据库-->
       <bean id="dataSource" class="com.alibaba.druid.pool.DruidDataSource" init-method="init" destroy-</pre>
method="close">
            <!-- 基本属性 url、user、password -->
            cproperty name="driverClassName" value="${jdbc driverClassName}"/>
            cproperty name="url" value="${jdbc_url}"/>
            cproperty name="username" value="${jdbc username}"/>
            <!-- 配置初始化大小、最小、最大 -->
            property name="initialSize" value="10"/>
            cproperty name="minIdle" value="10"/>
            <!-- 配置获取连接等待超时的时间 -->
            <!-- 配置间隔多久才进行一次检测,检测需要关闭的空闲连接,单位是毫秒 -->
            cproperty name="timeBetweenEvictionRunsMillis" value="60000" />
            <!-- 配置一个连接在池中最小生存的时间,单位是毫秒 -->
            cproperty name="minEvictableIdleTimeMillis" value="300000" />
            property name="validationQuery" value="SELECT 'x'" />
            cproperty name="testWhileIdle" value="true" />
            property name="testOnBorrow" value="false" />
            property name="testOnReturn" value="false" />
           <!-- 打开PSCache,并且指定每个连接上PSCache的大小 如果用Oracle,则把poolPreparedStatements配置为
true, mysql可以配置为false。-->
            cproperty name="poolPreparedStatements" value="false" />
            <!-- 配置监控统计拦截的filters -->
            property name="filters" value="wall,stat" />
      </bean>
      <!-- spring和MyBatis完美整合,不需要mybatis的配置映射文件 -->
      <bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
            <!-- 自动扫描mapping.xml文件 -->
           cproperty name="mapperLocations" value="classpath:org/ssm/dufy/mapper/*.xml"></property>
      </bean>
       <!-- DAO接口所在包名, Spring会自动查找其下的类,自动扫描了所有的XxxxMapper.xml对应的mapper接口文件,只要
Mapper接口类和Mapper映射文件对应起来就可以了-->
      <bean class="org.mybatis.spring.mapper.MapperScannerConfigurer">
            cproperty name="basePackage" value="org.ssm.dufy.dao" />
            cproperty name="sqlSessionFactoryBeanName" value="sqlSessionFactory">
      </bean>
     <!-- (事务管理)transaction manager, use JtaTransactionManager for global tx -->
<!-- 配置事务管理器 -->
      <bean id="transactionManager"</pre>
            class="org.springframework.jdbc.datasource.DataSourceTransactionManager">
            cproperty name="dataSource" ref="dataSource" />
      </bean>
      <!--== 事务配置 End =====
      <!-- 配置基于注解的声明式事务 -->
      <!-- enables scanning for @Transactional annotations -->
      <tx:annotation-driven transaction-manager="transactionManager" />
```

#### 测试代码,两种方式:

#### 测试1:

```
package org.ssm.dufy.service;
import org.junit.Test;
import org.junit.runner.RunWith;
{\tt import org.springframework.beans.factory.annotation.Autowired;}
import org.springframework.test.context.ContextConfiguration;
import org.springframework.test.context.junit4.SpringJUnit4ClassRunner;
import org.ssm.dufy.entity.User;
*配置spring和junit整合, junit启动时加载springIOC容器 spring-test, junit
@RunWith (SpringJUnit4ClassRunner.class)
// 告诉junit spring配置文件
@ContextConfiguration({ "classpath:applicationContext.xml"})
public class IUserServiceTest {
    @Autowired
    public IUserService userService;
    @Test
   public void getUserByIdTest(){
        User user = userService.getUserById(1);
       System.out.println(user.getUserName());
```

#### 测试2:

```
package org.ssm.dufy.service;
import org.springframework.context.ApplicationContext;
import org.springframework.context.support.ClassPathXmlApplicationContext;
import org.ssm.dufy.entity.User;

public class IUserServiceTest2 {

    public static void main(String[] args) {
        ApplicationContext application = new

    ClassPathXmlApplicationContext("applicationContext.xml");

        IUserService uService = (IUserService) application.getBean("userService");
        User user = uService.getUserById(1);
        System.out.println(user.getUserName());
    }
}
```

# 5:整合SpringMVC,添加配置,创建jsp

SpringMVC需要的依赖在pom.xml中已经加上了,现在需在Web项目中的web.xml中添加启动SpringMVC启动配置和Spring整合SpringMVC的配置了。

新增如下两个文件:

```
10
applicationContext.xml
                                                        114. 配置Spring+Springmvc配置文件
   idbc.properties
                                                        12

    Iogback.xml

                                                        135: 配置web.xml, 创建jsp
   x spring-mvc.xml
                                                        14
                                                        156.启动web服务,测试
   spring-mybatis.xml
> 🥞 src/test/java
                                                        16
                                                        177: 编写Dao-配置xml-Service-Contorller-

src/test/resources

> Mayen Dependencies
                                                        18
                                                        198: 测试业务代码
⇒ Maria JRE System Library [JavaSE-1.7]
> 📥 JUnit 4
> 🗁 logs

√ № src

                                                       Problems @ Javadoc B Declaration @ Search Declaration
  🗸 🔑 main
                                                       No operations to display at this time.
   v 🍃 webapp
     v 🍃 jsp
           error.jsp
           showName.jsp
         static
         web.xml
       index.jsp
   test 🍃
> 🗁 target
 db.sql
```

#### spring-mvc.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"</pre>
 xmlns:p="http://www.springframework.org/schema/p"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 xmlns:context="http://www.springframework.org/schema/context"
 xmlns:mvc="http://www.springframework.org/schema/mvc"
 xsi:schemaLocation="
   http://www.springframework.org/schema/beans
   http://www.springframework.org/schema/beans/spring-beans-3.2.xsd
   http://www.springframework.org/schema/context
   http://www.springframework.org/schema/context/spring-context-3.2.xsd
   http://www.springframework.org/schema/mvc
   http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd">
   <!-- 扫描controller (controller层注入) -->
   <context:component-scan base-package="org.ssm.dufy.web" use-default-filters="false">
       <context:include-filter type="annotation"</pre>
expression="org.springframework.stereotype.Controller"/>
   </context:component-scan>
    <mvc:annotation-driven />
     <!-- 内容协商管理器 -->
   <!--1、首先检查路径扩展名 (如my.pdf) ; 2、其次检查Parameter (如my?format=pdf) ; 3、检查Accept Header-->
   <bean id="contentNegotiationManager"</pre>
class="org.springframework.web.accept.ContentNegotiationManagerFactoryBean">
       <!-- 扩展名至mimeType的映射,即 /user.json => application/json -->
       cproperty name="favorPathExtension" value="true"/>
       <!-- 用于开启 /userinfo/123?format=json 的支持 -->
       property name="parameterName" value="format"/>
       <!-- 是否忽略Accept Header -->
       cproperty name="ignoreAcceptHeader" value="false"/>
       cproperty name="mediaTypes"> <!--扩展名到MIME的映射; favorPathExtension, favorParameter是true时</pre>
起作用 -->
               json=application/json
               xml=application/xml
               html=text/html
           </value>
       </property>
       <!-- 默认的content type -->
       cproperty name="defaultContentType" value="text/html"/>
   </bean>
   <!-- 当在web.xml 中 DispatcherServlet使用 <url-pattern>/</url-pattern> 映射时,能映射静态资源 -->
   <mvc:default-servlet-handler />
    <!-- 静态资源映射 -->
    <mvc:resources mapping="/static/**" location="/WEB-INF/static/"/>
   <!-- 对模型视图添加前后缀 -->
    <bean id="viewResolver"</pre>
class="org.springframework.web.servlet.view.InternalResourceViewResolver"
     p:prefix="/WEB-INF/jsp/" p:suffix=".jsp"/>
```

</beans>

#### web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
 <display-name>SSM-DEMO</display-name>
 <!-- 读取spring配置文件 -->
   <context-param>
       <param-name>contextConfigLocation</param-name>
       <param-value>classpath:applicationContext.xml</param-value>
   </context-param>
   <!-- 设计路径变量值
   <context-param>
       <param-name>webAppRootKey</param-name>
       <param-value>springmvc.root</param-value>
   </context-param>
   <!-- Spring字符集过滤器 -->
   <filter>
       <filter-name>SpringEncodingFilter</filter-name>
       <\!filter-class\!\!>\! org.springframework.web.filter.CharacterEncodingFilter<\!/filter-class>
       <init-param>
           <param-name>encoding</param-name>
           <param-value>UTF-8</param-value>
       </init-param>
       <init-param>
           <param-name>forceEncoding</param-name>
           <param-value>true</param-value>
       </init-param>
   </filter>
   <filter-mapping>
       <filter-name>SpringEncodingFilter</filter-name>
       <url-pattern>/*</url-pattern>
   </filter-mapping>
   <!-- 添加日志监听器 -->
   <context-param>
       <param-name>logbackConfigLocation</param-name>
       <param-value>classpath:logback.xml</param-value>
   </context-param>
       class>ch.qos.logback.ext.spring.web.LogbackConfigListener
   </listener>
   stener>
       tener-class>org.springframework.web.context.ContextLoaderListener
   </listener>
   <!-- springMVC核心配置 -->
   <servlet>
       <servlet-name>dispatcherServlet</servlet-name>
       <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
       <init-param>
           <param-name>contextConfigLocation</param-name>
           <!--spingMVC的配置路径 --:
           <param-value>classpath:spring-mvc.xml</param-value>
       </init-param>
       <load-on-startup>1</load-on-startup>
   </servlet>
   <!-- 拦截设置 -->
   <servlet-mapping>
       <servlet-name>dispatcherServlet</servlet-name>
       <url-pattern>/</url-pattern>
   </servlet-mapping>
 <welcome-file-list>
   <welcome-file>index.jsp</welcome-file>
 </welcome-file-list>
</web-app>
```

# 新增index.jsp文件

```
<%@ page contentType="text/html; charset=utf-8"%>
<!doctype html>
<html>
<body>
<h2>Hello World!</h2>
</body>
</html>
```

6.启动web服务,测试

将上面的项目添加到Tomcat中,启动,控制台没有报错,并在地址栏访问,http://localhost:8080/ssm。页面显示 Hello World! 项目配置ok!

7:编写Controller,和对应得业务界面

**新增UserController** ,通过参数传递uid获取用户,若用户存在,跳转到showName.jsp ,若用户不存在,则跳转到error.jsp,并返回提示信息!

```
package org.ssm.dufy.web;
import javax.servlet.http.HttpServletRequest;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RequestParam;
import org.springframework.web.servlet.ModelAndView;
import org.ssm.dufy.entity.User;
import org.ssm.dufy.service.IUserService;
@Controller
public class UserController {
   private IUserService userService;
    @RequestMapping(value="/showname", method=RequestMethod.GET)
   public String showUserName(@RequestParam("uid") int uid,HttpServletRequest request,Model model){
        System.out.println("showname");
       User user = userService.getUserBvId(uid);
       if(user != null) {
           request.setAttribute("name", user.getUserName());
           model.addAttribute("mame", user.getUserName());
           return "showName";
        request.setAttribute("error", "没有找到该用户! ");
        return "error";
```

# Jsp内容如下:

```
47
48
49

src/main/resources

                                                                                                                                                                                               <mvc:resources mapping="/static/**" location="/WEB-INF/static/"/>

    applicationContext.xml
    jdbc.properties

         <!-- 对模型视图添加前后缀 -->

    spring-mvc.xml
    spring-mybatis.xml
    spring-mybatis.xml
    src/test/java
    src/test/jresources
    Maven Dependencies
    M. RRE System Library [JavaSE-1.7]
    M. Julit 4

                                                                                                                                                                                                   xbean id="vtewResotver" class="org.springframewonk.web.servlet.view.InternalResource
p:prefix="/WEB-INF/jsp/" p:suffix=".jsp"/>
                                                                                                                                                                                       /beans>
     Design Source

Problems Baudoc B Declaration Search Console B As Unit To Progress & Debug 2-Call Hierarchy

B: Starting ProtocolHandler ["http-bio-8088"]

In 20, 2017 8:44:54 Theorg.apache.coyote.AbstractProtocol start

B: Starting ProtocolHandler ["ajp-bio-8089"]

In 20, 2017 8:44:54 Theorg.apache.catalina.startup.Catalina start

Existence Server startup in 5852 ms

showname

2017-02-20 20:45:01.886 DEBUG[http-bio-8080-exec-3]org.mybatis.spring.SqlSessionUtils.getSql

2017-02-20 20:45:01.952 DEBUG[http-bio-8080-exec-3]org.mybatis.spring.SqlSessionUtils.getSql

2017-02-20 20:45:01.952 DEBUG[http-bio-8080-exec-3]org.mybatis.spring.transaction.SpringMana

2017-02-20 20:45:01.954 DEBUG[http-bio-8080-exec-3]org.ssm.dufy.dao.IUserDao.selectbyPrimary
      v 🐎 main
            ~ @ webapp
                     jsperror.jspshowNar
test 
arget
```

# showName.jsp

# error.jsp

</body> </html> 三: 遇到的问题 1:找不到UserService, 报错 可能是包扫描路径的问题,检查一下Service是否在扫描的范围内 2: jsp接收不到request.setAttribute("","");内容 查资料说是因为 JSP的版本问题,可以在Jsp 上面添加 <%@ page isELIgnored="false" 或者修改web.xml,添加version版本! <?xml version="1.0" encoding="UTF-8"?> <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</pre> xmlns="http://java.sun.com/xml/ns/javaee" xmlns:web="http://java.sun.com/xml/ns/javaee/webapp\_2\_5.xsd" si:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/webapp\_3\_0.xsd" id="WebApp ID" version="3.0" metadata-complete="true"> 四: 心得总结 或许这些都是站在别人的基础的搭建的,但是看一篇SSM的搭建文章可能很快,看完觉得自己懂了,但是我建议一定要 自己去搭建一次,看一遍,和动手做一遍完全是两个概念! 五:参考文章 SSM框架——详细整合教程(Spring+SpringMVC+MyBatis) Mybatis3.x与Spring4.x整合 附: 项目源代码: https://github.com/dufyun/learn-tech-collection/tree/master/ssm 欢迎访问我的csdn博客,我们一同成长! "不管做什么, 只要坚持下去就会看到不一样! 在路上, 不卑不亢!" http://blog.csdn.net/u010648555 本文为博主-阿飞(dufy)-原创文章,未经博主允许可转载,但请标明出处,谢谢! 分类: Spring&SpringMVC 标签: SSM框架, SpringMVC, Spring, Mybatis 好文要顶 ( 关注我 ) ( 收藏该文 )



2

« 上一篇: Navicat for MySQL: 快捷键整理

» 下一篇: java集合系列——java集合概述 (一)

posted on 2017-02-20 20:58 阿飞(dufyun) 阅读(77613) 评论(14) 编辑 收藏

#### 评论

# #1楼

哥,你依赖里面两个spring-webmvc,maven能自动下载才怪你,你坑人啊

支持(0) 反对(0)

2017-12-17 16:32 | LAOK23

# #2楼[楼主]